Kotlin - Constructors

A Kotlin constructor is a special member function in a class that is invoked when an object is instantiated. Whenever an object is created, the defined constructor is called automatically which is used to initialize the properties of the class.

> Every Kotlin class needs to have a constructor and if we do not define it, then the compiler generates a default constructor.

A Kotlin class can have following two type of constructors:

- Primary Constructor
- Second Constructors

A Kotlin class can have a primary constructor and one or more additional secondary constructors. The Kotlin primary constructor initializes the class, whereas the secondary constructor helps to include some extra logic while initializing the class.

Kotlin Primary Constructor

The primary constructor can be declared at class header level as shown in the following example.

```
class Person constructor(val firstName: String, val age: Int) {
   // class body
}
```

The **constructor** keyword can be omitted if there is no annotations or access modifiers specified like public, private or protected..

```
class Person (val firstName: String, val age: Int) {
   // class body
}
```

In this example, we have declared properties through the **val** keyword to make them read-only. These properties could be defined using keyword **var** if you need to change their values at later point in time.

Initializer Block

The primary constructor cannot contain any code. Initialization code can be placed in initializer blocks prefixed with the **init** keyword. There could be more than one **init** blocks and during the initialization of an instance, the initializer blocks are executed in the same order as they appear in the class body, interleaved with the property initializers:

Following is an example with a usage of initializer block:

```
class Person (val _name: String, val _age: Int) {
   // Member Variables
   var name: String
   var age: Int

   // Initializer Block
   init {
      this.name = _name
      this.age = _age
      println("Name = $name")
      println("Age = $age")
   }
}

fun main(args: Array<String>) {
   val person = Person("Zara", 20)
}
```

When you run the above Kotlin program, it will generate the following output:

```
Name = Zara
Age = 20
```

Default Values

Kotlin allows to initialize the constructor parameters with some default values. Following is a working example for the same:

```
class Person (val _name: String, val _age: Int=20) {
   // Member Variables
   var name: String
   var age: Int

   // Initializer Block
   init {
      this.name = _name
      this.age = _age
      println("Name = $name")
      println("Age = $age")
   }
}

fun main(args: Array<String>) {
   val zara = Person("Zara")
   val nuha = Person("Nuha", 11)
}
```

When you run the above Kotlin program, it will generate the following output:

```
Name = Zara
Age = 20
Name = Nuha
Age = 11
```

Kotlin Secondary Constructor

As mentioned earlier, Kotlin allows to create one or more secondary constructors for your class. This secondary constructor is created using the **constructor** keyword. It is required whenever you want to create more than one constructor in Kotlin or whenever you want to include more logic in the primary constructor and you cannot do that because the primary constructor may be called by some other class.

Example

Take a look at the following example, here we have created a secondary constructor to implement the above example once again:

```
class Person{
   // Member Variables
   var name: String
   var age: Int

   // Initializer Block
   init {
      println("Initializer Block")
   }

   // Secondary Constructor
   constructor ( _name: String, _age: Int) {
      this.name = _name
      this.age = _age
      println("Name = $name")
      println("Age = $age")
   }
}
fun main(args: Array<String>) {
   val zara = Person("Zara", 20)
}
```

When you run the above Kotlin program, it will generate the following output:

```
Initializer Block
Name = Zara
Age = 20
```

Secondary constructor do not allow to use **val** or **var** with secondary constructor parameters. Now let's see one example with two secondary constructors:

```
class Person{
   // Member Variables
   var name: String
   var age: Int
   var salary:Double

   // First Secondary Constructor
   constructor ( _name: String, _age: Int) {
      this.name = _name
      this.age = _age
      this.salary = 0.00
      println("Name = $name")
      println("Age = $age")
   }

   // Second Secondary Constructor
   constructor ( _name: String, _age: Int, _salary: Double) {
      this.name = _name
      this.age = _age
      this.salary = _salary
      println("Name = $name")
      println("Age = $age")
      println("Salary = $salary")
   }
}

fun main(args: Array<String>) {
   val nuha = Person("Nuha", 12)
   val zara = Person("Zara", 20, 2000.00)
}
```

When you run the above Kotlin program, it will generate the following output:

```
Name = Nuha
Age = 12
Name = Zara
Age = 20
Salary = 2000.0
```

Quiz Time (Interview & Exams Preparation)

Q 1 - We can define N number of constructors in Kotlin program:

A - Yes

B - No

Q 2 - Which keyword is used to define a Kotlin constructor:

A - init

B - constructor

C - Constructor

D - All the bove

Q 3 - Kotlin allows to set default values for the constructor parameters.

A - True

B - False

Q 4 - Second constructor does not allow to use data types alongwith its parameters.

A - True

B - B